Engineered Science
DOI: http://dx.doi.org/10.30919/es1668

JiuCuo: PacBio HiFi Read Correction Method using Preassembled Contigs based on Deep Image Processing

Jiwen Liu,[1,#] Mingfei Pan,[1,#] Hongbin Wang,[1,#] Hang Zhang[2,*] and Ergude Bao[1,*]

Abstract

In biomedical field, as a fundamental technology to obtain genome sequence towards disease treatment, the PacBio HiFi sequencing technology emerged these years has greatly benefited genome analysis especially genome assembly. Even if the HiFi reads contains only 1% base errors, correction is crucial before assembly. Most of the current assembly methods contain an effective and efficient error correction module, but there is still some space for improvement: all-versus-one read alignment could be applied for accuracy, base errors and SNPs could be distinguished, and adapters could also be removed. To achieve these improvements, we design a HiFi read correction method called JiuCuo, which aligns the reads to preassembled contigs to do SNP-aware error correction and adapter removal. JiuCuo converts the aligned reads into images, then applies improved Inception-v4 model to distinguish base errors from SNPs, and also applies YOLO-v8 model combining DBSCAN and k-mer-based matching to detect adapters. In our tests, JiuCuo is accurate in SNP-aware error correction and adapter removal, and the JiuCuo corrected reads could be further assembled into contigs of higher quality than the preassembled ones. The JiuCuo software can be downloaded freely from https://github.com/liuj001/jiucuo.

Keywords: HiFi reads; Base error correction; Adapter removal; Deep image processing.
Received: 11 April 2025; Revised: 18 July 2025; Accepted: 28 July 2025.
Article type: Research article.

1. Introduction

Genome sequencing is the fundamental process for biomedical studies towards disease treatment, where the obtained genome sequence is the basis of gene-disease analysis and drug discovery. For example, not only does novel cancer drug discovery rely on genomic sequencing, but so does drug selection for individual patients. To obtain a genome's sequence, firstly, a sequencing machine is applied to generate the sequence fragments in a biological process; secondly, a correction algorithm is run to correct errors in the fragments in a computational process; thirdly, an assembly algorithm is also run to put the corrected fragments together to fit the complete genome sequence. The sequencing machine/technology evolves to generate more accurate sequence fragments, and one of the most recent and widely used is the PacBio HiFi sequencing technology. With this technology, the obtained read length has increased from several hundred to ten-thousands, and the error rate has dropped to 1%. As a result, the downstream genome assembly can benefit to a large extent. The reason is three-fold. (a) Repetitive regions in genome can only be assembled with reads spanning them, and the HiFi reads are sufficiently long. (b) Assembly quality is largely affected by base errors in the reads, and the HiFi reads have low error rate. (c) Haplotype resolved assembly needs read spanning multiple SNPs with low error rate to clearly distinguish the SNPs, and the HiFi reads also fulfill this requirement.

For the HiFi read assembly, a few assemblers have been developed and can obtain quality assembly results. First step of almost all these assemblers is error correction, even if the HiFi reads have as low as 1% base error rate. As stated in the Hifiasm paper:[1] the error correction can guarantee simple and clean assembly graph built from the reads, and thus quality contigs assembled. Most of the error correction methods (a) compress each homopolymer to a single read base and then (b) align the HiFi reads with each other for correction. In step (a), the homopolymer is consecutive bases of the same type (for example, AAAAA), and previous studies have demonstrated

[1] *Group of Interdisciplinary Information Sciences, School of Software Engineering, Beijing Jiaotong University, Beijing, 100044, China*
[2] *Institute of Engineering Thermophysics, Chinese Academy of Sciences, Beijing, 100044, China*
[*] *Email:* zhanghang@iet.cn (Hang Zhang); baoe@bjtu.edu.cn (Ergude Bao)
[#] *These authors contributed equally to this work.*

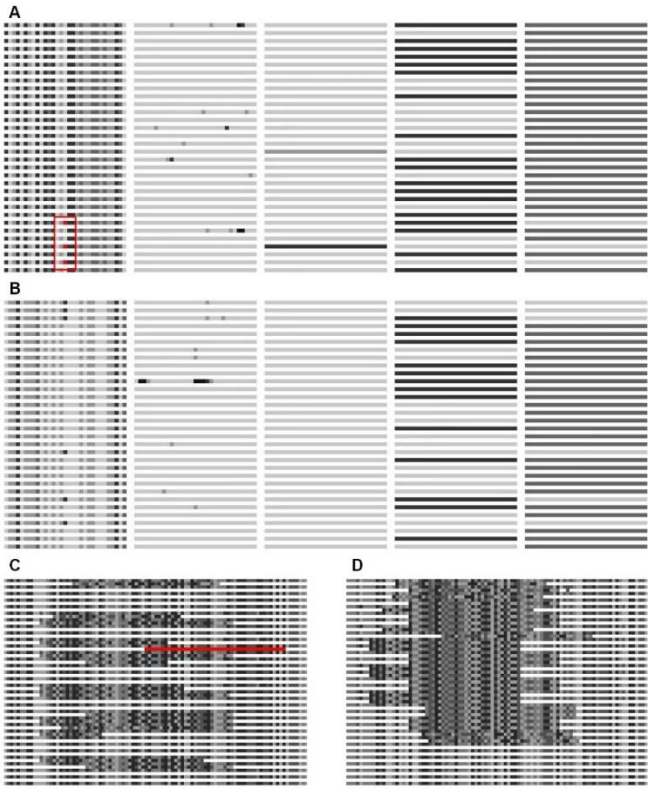

Fig. 1: Example images generated by JiuCuo from real HiFi reads. (A) Images from one spot of aligned *A. thaliana* reads, with three errors in a line (colored in red and highlighted with a red rectangle) similar to SNPs. (B) Images from another spot of aligned *A. thaliana* reads, with SNPs in the middle. (C) Image from one spot of human (HG002) reads, with an adapter (colored in red) similar to SVs. (D) Image from another spot of human HiFi reads, with SVs in the middle. Details of these images are described in section "Image generation".

that HiFi reads may contain incorrect number of bases in some homopolymers, so compressing the bases can correct such errors.[2] Even if information loss occurs with the compression, it is beneficial for the downstream assembly process and the compressed bases can be recovered at last. Then in step (b), the alignment of HiFi reads enables comparing bases in different reads but from the same genome position, and majority vote approach can be applied to detect and correct base errors. Specifically, HiCanu compresses homopolymers and aligns the HiFi reads with each other for correction;[2] Hifiasm directly aligns the HiFi reads with each other for correction with concern about distinguishing base errors from SNPs;[1] mdBG compresses homopolymers, abstracts each HiFi read to a set of partially ordered short sequences called minimizers and aligns these minimizers with each other for correction;[3] LJA compresses homopolymers, constructs a compressed de Bruijn graph from the HiFi reads and aligns the reads to the graph for correction;[4] Verkko not only compresses homopolymers but also short repeats (for example, ACACAC to AC) and aligns the HiFi reads with each other for correction.[5] It is worthwhile to note that, the "HiFi read correction" here is different from the "subread correction for HiFi read generation", which is studied in the DeepConsensus paper.[6]

The current error correction methods for HiFi reads are efficient and effective, but there are still some challenges to address. (a) Most of the methods rely on all-versus-all read alignment, but aligning the reads to a "single target" such as reference genome or de Bruijn graph would be more accurate and beneficial for error correction (LJA does this). (b) Most of the methods do not concern about distinguishing base errors from SNPs, and may either over-correct some SNPs or under-correct errors near SNPs (Hifiasm and HiCanu are exceptions). (c) Most of the methods do not try to remove adapters inserted in the reads (HiCanu is an exception). To address these challenges, we propose JiuCuo, a PacBio HiFi read correction method using preassembled contigs based on deep image processing. Following HALC[7] and FLAS,[8] this is the third PacBio long read correction tool designed by us. The first two target at correcting PacBio subreads, while JiuCuo target at PacBio HiFi reads. The design philosophy of JiuCuo is as below.

• JiuCuo relies on de novo contigs preassembled from initial HiFi reads as input. Before de novo assembly, to correct the HiFi reads, no other information is available, and the only way is to align them with each other, and this is what most assemblers do; during the de novo assembly, the constructed assembly graph can be used as the "single target" as mentioned above for correction, and this is what LJA does; after the de novo assembly, the assembled contigs can be used as the "single target" for correction, and this is what JiuCuo does. The error corrected HiFi reads with JiuCuo could be reassembled into contigs of higher quality than the preassembled ones. This addresses challenge (a) as described in the previous paragraph.

• JiuCuo is based on deep image processing. Recent studies have demonstrated it effective and accurate to convert read alignment into images and use deep learning methods for various analysis. As a representative work, DeepVariant can achieve very high SNP detection accuracy.[9] The reason of effectiveness is, deep image processing does not rely on curated rules so it is able to make accurate and robust data analysis in complex scenarios. Following this direction, we let JiuCuo convert aligned HiFi reads into images, and then make SNP-aware error correction and remove adapters with modified deep image processing models. Specifically, to achieve SNP-aware error correction, JiuCuo locates a superset of base errors with SNPs, and then distinguishes the errors from SNPs so that the former can be corrected. In this process, JiuCuo locates the superset simply by comparing each base with surrounding bases, and then distinguishes the errors from SNPs with improved Inception- v4 model[10] following DeepVariant. To achieve adapter removal, JiuCuo locates a superset of adapters with SVs, and then refines the set to correctly remove the adapters. In this process, JiuCuo locates the superset with YOLO-v8 model,[11] and then refines the

detection results with their consensus and sequence information to obtain the final adapter set. This addresses challenges (b) and (c) as described in the previous paragraph. Fig. 1 presents example images generated by JiuCuo from real HiFi reads. In the examples, base errors (Fig. 1(A)) could be highly similar to SNPs (Fig. 1(B)), and an adapter mixed with SVs (Fig. 1(C)) could also be highly similar to pure SVs (Fig. 1(D)), showing necessity to use deep image processing models to make distinction.

In summary, we have the following contributions in this study.

• We propose to preassemble the HiFi reads into contigs and align the reads to the contigs for correction.

• We convert the aligned HiFi reads into images and modify deep image processing models to make SNP-aware error correction and adapter removal.

• We improve Inception-v4 model by replacing its MaxPool and AveragePool modules by eDSCWPool and eMPool,[12,13] respectively, which are more sensitive to distinguish aggregated base errors from SNPs.

• We combine YOLO-v8 model with DBSCAN clustering algorithm and k-mer based sequence matching approach, to accurately detect adapters.

• In our tests, JiuCuo is accurate in SNP-aware error correction and adapter removal, and the JiuCuo corrected human reads could be assembled into contigs with NGA50 improvement from 64.74-88.85 mbp to 75.13-93.65 mbp.

2. Methods

2.1 Overview

The JiuCuo method is inputted with HiFi reads and preassembled contigs from the reads, and it does SNP-aware error correction and adapter removal in following steps to correct the HiFi reads. JiuCuo calls Minimap2, Inception-v4 (with modifications ourselves), YOLO-v8 and DBSCAN in necessary places. The choice of Inception-v4 follows DeepVariant and v4 is Inception's latest version. The choice of YOLO-v8 is because YOLO is the most widely used object detection method and v8 is the latest version. Fig. 2 is an illustration of the JiuCuo method.

1. Image generation. Align the HiFi reads to the preassembled contigs with Minimap2[14] and based on the alignment results, generate a set of four basic images each emphasizing one aspect of the alignment.

2. SNP-aware error detection. Scan all columns of the generated images from step 1. If mismatches exist in a column, generate a set of five sub-images. Input the five sub-images into an improved Inception-v4 model to determine if the mismatches in the column are due to base errors or SNPs. Record all the determined errors and additional single insertions/deletions.

3. Adapter detection. Input the set of four basic images from step 1 into YOLO-v8 model to obtain candidate adapters, and apply DBSCAN algorithm and k-mer based matching approach to determine if a candidate adapter is an adapter or SV. Record all the determined adapters.

4. Correction with detected errors and adapters. Correct the recorded base errors from step 2, and remove the recorded adapters from step 3.

2.2 Image generation

We use Minimap2 with default settings to align the HiFi reads to the preassembled contigs. The aligned contigs are restricted to primary contigs, because the reads from homologous chromosomes need to be aligned and compared in downstream processing, and also because primary contigs are usually assembled with higher quality than alternative ones. Based on the alignment results, we generate four basic images of the same size: base-type image, base-quality image, alignment quality image and alignment-orientation image. Besides alignment information, the base-type image stores base type information, the base-quality image stores base quality information, the alignment-quality image stores alignment quality information, and the alignment-orientation image stores alignment orientation information. These images are inspired by DeepVariant,[9] but have quite a few differences. Below are details of the images.

• **Base-type image.** Each column of this image represents a contig base, so the image width w is equal to number of all the contig bases. Two adjacent rows represent an aligned read with the first representing ordinary bases or deletions and the second representing insertions, so the image height h is equal to twice alignment depth of the reads. Each pixel in this image represents the corresponding read base with grayscale value 5 representing A, 6 representing T, 7 representing C, 8 representing G, and 9 representing a missing base.

• **Base-quality image.** Each column and two adjacent rows have the same meaning with the base-type image, so it is also a w×h image. Each item in this image represents quality of the corresponding read base with grayscale value 5 representing quality value over 55, 6 representing the value between 41 and 54, 7 representing the value between 37 and 40, and 8 representing the value below 36.

• **Alignment-quality image.** Each column and two adjacent rows have the same meaning with the base-type image, so it is also a w×h image. Each item in this image represents alignment quality of the corresponding read with grayscale value 5 representing alignment quality value over 31, 6 representing the value between 21 and 30, 7 representing the value between 11 and 20, and 8 representing the value below 10. The alignment quality value for each read can be obtained directly from MAPQ column of the read in alignment file.

• **Alignment-orientation image**. Each column and two adjacent rows have the same meaning with the base-type image, so it is also a w×h image. Each item in this image represents alignment orientation of the corresponding read with grayscale value 5 representing reverse alignment, and 8 representing forward alignment.

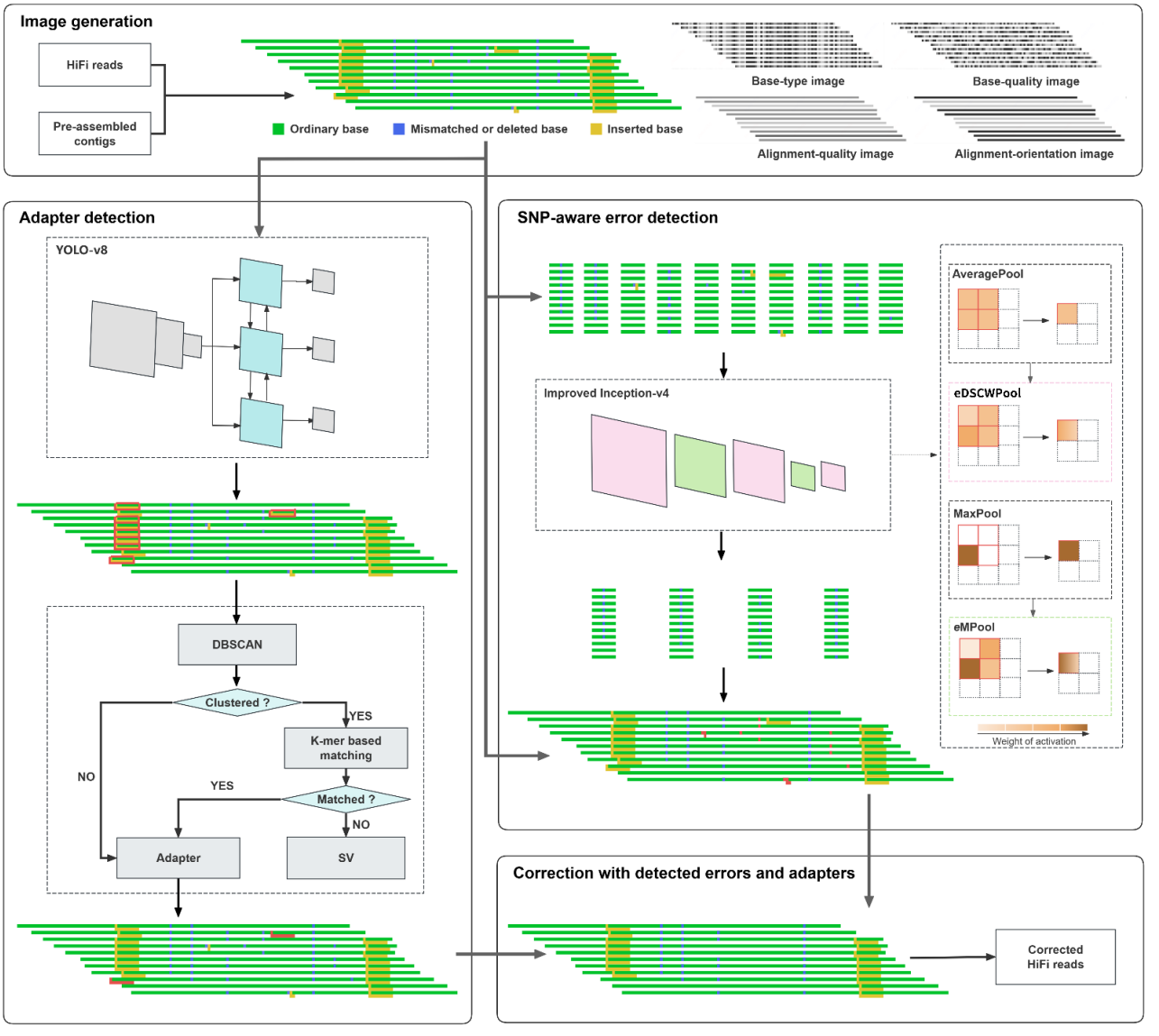

Fig. 2: Illustration of the JiuCuo method. In image generation, HiFi reads are aligned to contigs to generate a set of four basic images (among the four, the base-type image is colored, simplified and used in this illustration with an ordinary base colored green, a mismatch/deletion colored blue and an insertion colored yellow). In SNP-aware error detection, a few columns in the base-type image contain mismatches, so each of them generates a set of five sub-images (among each set of five sub-images, the base-type sub-image is use in this illustration). Each set of sub-images is inputted into an improved Inception-v4 model with eDSCWPool and eMPool to distinguish base errors from SNPs. The detected errors are recorded and indicated in red. In adapter detection, the set of four basic images are inputted into YOLO-v8 to obtain candidate adapters indicated with red rectangles. The candidate adapters are further processed with DBSCAN and k-mer based matching to exclude SVs. The detected adapters are recorded and indicated in red. In correction with detected errors and adapters, the recorded errors and adapters are removed from the HiFi reads.

It is worthwhile to note that, since the reads are aligned to preassembled contigs, their order is completely decided by the contigs, and the generated images are fixed. That is, no matter how the reads are sorted, after alignment to the contigs, their order and the generated images do not change. It is also worthwhile to note that, we choose these grayscale values in JiuCuo, simply because the generated images styles are consistent with DeepVariant. These values can be changed as long as different images, and this does not hurt or optimize the results. As long as these values are decided, they are used as a protocol in both training and prediction, so there will be no overfitting issue.

2.3 SNP-aware error detection

We scan the base-type image column by column, and if the ith column contains mismatches with a different grayscale, cut w′ columns with the ith column in the center from the base-type image. The obtained image is a w′×h base-type sub-image. In addition, we also cut the corresponding columns from the base-quality image, alignment-quality image and alignment-

orientation image, obtaining w′×h base-quality sub-image, alignment-quality sub-image and alignment-orientation sub-image. Besides these four sub-images, we generate the fifth mismatch sub-image. Each column and two adjacent rows of the mismatch sub-image have the same meaning with the base type sub-image, so it is also a w′×h sub-image. Each row in this sub-image represents existence of mismatch in the ith column with grayscale 5 representing existence, and 8 representing not existence. This sub-image is essentially enhancement of the mismatch column in the base-type image. With the set of five sub-images, we apply an improved Inception-v4 model[10] to determine if the mismatches in the ith column are due to base errors or SNPs. Then we record the determined base errors and additional single insertions/deletions.

The Inception-v4 model is used and improved by replacing the AveragePool and MaxPool with eDSCWPool and eMPool, respectively.[12,13] We improve Inception-v4 in this way, because some base errors appear in a line quite similar to SNPs, and it is necessary to retain sufficient multi-scale details in the sub-images during processing. It is worthwhile to note that, though Inception-v4 itself contains designs to retain multi-scale details, and more complicated techniques such as feature pyramid can also be incorporated, we find updating the pooling modules is quite effective and easy. Fig. S1 presents an illustrative image with SNPs, and the processing results with AveragePool, MaxPool, eDSCWPool and eMPool.

Specifically, given a pooling field feature set R, the formula of AveragePool is as below, which averages all the features $a_j \in$ R as the pooling result.

$$\tilde{a} = \sum_{a_j \in R} \frac{a_j}{|R|} \quad (1)$$

This may lose small-scale features. Differently, the formula of eDSCWPool is as below, which calculates dice similarity coefficient (DSC) distance between each feature a_j and the average $\bar{a}$, and assigns large weight to the a_j close to $\bar{a}$.

$$\tilde{a} = \sum_{a_j \in R} \frac{e^{DSC(\bar{a}, a_j)}}{\sum_{a_k \in R} e^{DSC(\bar{a}, a_k)}} \cdot a_j \quad (2)$$

This can average the features while preserving information of small-scale features in the result. Here, The DSC distance is calculated as below.

$$DSC(\bar{a}, a_j) = \frac{2\sum_i (\bar{a}(i) \cdot a_j(i))}{\sum_i \bar{a}(i) + \sum_i a_j(i)} \quad (3)$$

On the other hand, the formula of MaxPool is as below, which chooses the feature a_j of max value as the pooling result.

$$\tilde{a} = \max_{a_j \in R} a_j \quad (4)$$

This may also lose small-scale features. Differently, the formula of eMPool is as below, which assigns large weight to the a_j with large value.

$$\tilde{a} = \sum_{a_j \in R} \frac{e^{a_j}}{\sum_{a_k \in R} e^{a_k}} \cdot a_j \quad (5)$$

This can find the max feature as the result while preserving information of small-scale features.

2.4 Adapter detection

We detect adapters from the generated four basic images with a combination of commonly used deep learning model and algorithms: YOLO-v8 model,[11] DBSCAN clustering algorithm and k-mer based sequence matching approach. The YOLO-v8 model and DBSCAN algorithm use consensus information with sequence information in the input images to detect candidate adapters and then determine some adapters. Here, the candidate adapters are adapter-like insertions and the determined adapters are single insertions not clustered with each other, so the remaining candidate adapters are adapter-like insertions clustered with each other. Then the k-mer based matching approach further uses sequence information for the remaining candidate adapters to determine more adapters.

Specifically, first, we apply YOLO-v8 to detect candidate adapters, and the detected adapters are mingled with SVs of similar sequence composition. Although not frequent, such SVs exist in the reads, and need to be further excluded. Second, we apply DBSCAN to cluster the detected candidate adapters based on column numbers of their first bases. Two candidate adapters p_i and $p_{i'}$ of the ith and i′th columns, respectively, are clustered, if their distance is not larger than diameter_size, and a cluster is valid if number of the clustered candidate adapters exceeds cluster_size. That is, each cluster c fulfills the following formula.

$$c = \{\forall p_i, p_{i'} \in c \mid |i - i'| \leq diameter\_size \wedge |c| > cluster\_size\} \quad (6)$$

Not clustered candidate adapters are recorded as determined adapters, and the clustered candidate adapters need to be further investigated. Third, we apply k-mer based sequence matching to align the candidate adapters with adapter templates, such as AAAAAAAAAAAAAAAAATTAACG GAGGAGGAGGA. With a fixed k-mer size k_size, a candidate adapter is recorded as a determined adapter, if ratio of matched k-mers exceeds identity_value. That is, each determined adapter p compared with template t fulfills the following formula, where K(p, k_size) is p's k-mer set of k_size.

$$\frac{|K(p, k\_size) \cap K(t, k\_size)|}{|K(t, k\_size)|} > identity\_value \quad (7)$$

In this way, we record all the determined adapters. It is worthwhile to note that, since the basic images are usually very large in width, we split them into chunks for adapter detection.

2.5 Correction with detected errors and adapters

With the recorded base errors and adapters, we remove the errors by comparing their adjacent bases in the base-type images, and also remove the adapters. If an adapter is located in the middle of a read, we remove the adapter and split the

read into two before and after the adapter; if an adapter is located in a read end, we simply remove the adapter with the end.

2.6 Training and implementation details

To train the improved Inception-v4 model, we use the labeled SNP dataset from human (HG001) with adaptation to JiuCuo inputted format.[9] Although more diverse datasets could lead to higher accuracy, the human dataset is sufficient in our tests, because SNP information does not differ much across species. One tenth of the training dataset is used for validation. Inception-v4 is trained with batch size 8 and learning rate determined by stochastic gradient descent (SGD) optimizer with initial value 0.0001, and the training is stopped when loss is below 0.1 and accuracy is above 95% on the validation set.

To train the YOLO-v8 model, we insert adapters to the HiFi reads from human (HG001) to form a training dataset, because no adapter dataset is available to the best of our knowledge. According to the HiFiAdaptFilt paper,[15] we insert adapters into 0.05% of the reads, and most of the adapters are inserted to the read ends. The inserted adapter has length between 40 and 45 or between 30 and 35, depending on the adapter template, and may contain 1-2 bases different from the template. One tenth of the training dataset is used for validation. YOLO-v8 is trained with batch size 16 and learning rate determined by Adam optimizer with initial value 0.001, and the training is stopped when loss is below 0.05 and accuracy is above 95% on the validation set. It is worthwhile to note that, the Inception-v4 and YOLO-v8 models are completely independent with different inputs and outputs, so independent and joint training of them do not have any difference in JiuCuo's performance. JiuCuo is developed in Python on Linux system. To comply with Inception-v4 and YOLO-v8's input requirements, they are fed with matrices and real images, respectively. JiuCuo can be run with a specified number of threads and each thread is allocated with a specified number of reads. In addition, it can either skip the base error correction or adapter removal step depending on specified command line option.

3. Experiments

3.1 Experimental design

3.1.1 Comparison with existing error correction methods

To assess performance of JiuCuo in base error correction, we choose three HiFi read datasets from species *A. thaliana*, *D. melanogaster* and human (HG002) of genome sizes 139 mbp, 125 mbp and 3 gbp, respectively. Genome coverages of these HiFi reads are approximately 100x, 125x and 50x, respectively. For each species, we correct base errors by correction modules in HiCanu,[2] Hifiasm[1] and mdBG.[3] We continue to assemble the reads into contigs, so we have three sets of preassembled contigs with HiCanu, Hifiasm and mdBG, respectively. Using each preassembled contig set, we correct the initial HiFi reads by JiuCuo. The settings of HiCanu, Hifiasm, mdBG and JiuCuo are all the defaults, while the HiCanu's *genomeSize* option does not have a default value and is set according to the sizes above. It is worthwhile to note that, HALC, FLAS and some other PacBio subreads correction methods are not compared, since they cannot correct HiFi reads. To evaluate the error correction results, we align the initial reads and corrected reads to the Hifiasm and HiCanu assembled haplotype-resolved contigs and use Hifieval.[16] The Hifiasm and HiCanu assembled contigs are used for evaluation, because of lack of perfect haplotype-resolved reference genomes, and as stated in the Hifieval paper: perfect assembly is not needed in order to sufficiently evaluate the read correction performance. For fair comparison, we do necessary homopolymer compressions before evaluation. Besides this test, we also analyze JiuCuo's scalability and running time in sections "Impact of error rate on correction results" and "Running time with increased threads" in Supplementary information.

3.1.2 Comparison with other adapter removal methods

To assess the performance of JiuCuo in adapter removal, we also use the same HiFi read datasets as above. As discussed in section "Introduction", HiCanu can remove adapters, but it does not report positions of the adapters, so we compare JiuCuo with CutAdapt[17] and HiFiAdapterFilt.[15] According to the experimental design above, we have three sets of preassembled contigs with HiCanu, Hifiasm and mdBG, respectively. Using each preassembled contig set, we remove adapters from the initial reads by JiuCuo. The settings of CutAdapt, HiFiAdapterFilt and JiuCuo are all the defaults. To evaluate the adapter removal results, since no ground truth of the existing adapters is available, we match and compare the removed adapters by the three methods.

3.1.3 Reassembly with JiuCuo

As discussed in section "Introduction", correction of HiFi reads could have positive effect on downstream assemblies. To assess JiuCuo's effect, we reassemble JiuCuo corrected HiFi reads. For each species, according to the experimental design above, we have three sets of JiuCuo corrected HiFi reads with base error correction and adapter removal, and they were obtained using preassembled contigs with HiCanu, Hifiasm and mdBG, respectively. For each set of JiuCuo corrected HiFi reads, we reassemble them with the corresponding genome assembler. For example, JiuCuo corrected HiFi reads using HiCanu preassembled contig set are reassembled with HiCanu. To evaluate the assembly results, we use QUAST which aligns the assembled contigs to the corresponding reference genomes for comparison.[18] In addition, we also use Mummer to generate the assembly plots.[19] Though the reference genomes are not haplotype-resolved, they are sufficient to evaluate JiuCuo's effect.

Table 1: Error correction results by various methods on *A. thaliana*, *D. melanogaster* and human (HG002) datasets.

Method	TPR	FNR	OCR	PDR	HDR	CDR
			(a) *A. thaliana* dataset			
HiCanu^c	72.44%	27.56%	2.89%	0.02‰	0.16‰	0.01‰
JiuCuo(C)	85.54%	14.46%	2.30%	0.01‰	0.19‰	0.01‰
Hifiasm^c	97.42%	2.58%	0.55%	0.04‰	3.42‰	0.03‰
JiuCuo(F)	98.80%	1.20%	1.07%	0.02‰	3.65‰	0.03‰
mdBG^c	99.49%	0.51%	84.75%	0.33‰	6.48‰	0.10‰
JiuCuo(M)	90.12%	9.88%	4.67%	0.09‰	4.49‰	0.04‰
			(b) *D. melanogaster* dataset			
HiCanu^c	69.16%	30.84%	4.99%	0.01‰	0.16‰	0.01‰
JiuCuo(C)	81.12%	18.88%	5.22%	0.02‰	0.20‰	0.01‰
Hifiasm^c	97.32%	2.68%	0.83%	0.05‰	2.91‰	0.01‰
JiuCuo(F)	97.56%	2.44%	1.51%	0.05‰	4.59‰	0.01‰
mdBG^c	99.01%	0.99%	90.36%	0.21‰	5.84‰	0.06‰
JiuCuo(M)	89.69%	10.31%	4.20%	0.05‰	3.68‰	0.02‰
			(c) human (HG002) dataset			
HiCanu^c	68.49%	31.51%	6.04%	0.00‰	0.21‰	0.00‰
JiuCuo(C)	79.65%	20.35%	5.78%	0.01‰	0.24‰	0.00‰
Hifiasm^c	87.11%	12.89%	1.42%	0.11‰	2.12‰	0.02‰
JiuCuo(F)	87.26%	12.74%	2.35%	0.05‰	2.68‰	0.03‰
mdBG^c	96.56%	3.44%	95.77%	0.40‰	7.14‰	0.08‰
JiuCuo(M)	88.62%	11.38%	14.29%	0.11‰	6.52‰	0.06‰

Note: HiCanu^c, Hifiasm^c, and mdBG^c represent the error correction modules of HiCanu, Hifiasm, and mdBG, respectively. JiuCuo(C), JiuCuo(F), and JiuCuo(M) represent JiuCuo using preassembled contigs from HiCanu, Hifiasm, and mdBG, respectively. In addition, mdBG's OCR is large because it does not need SNP-aware error correction in its design.

.

3.1.4 Impact of model improvement in SNP-aware error detection

As discussed in section "Method", improvement of the Inception-v4 model in JiuCuo could have positive impact on base error correction. To assess the impact of model improvement in distinguishing errors from SNPs, we directly compare the original Inception-v4 and JiuCuo in classifying sub-images with and without SNPs. We use a SNP dataset in consistency with the *A. thaliana* HiFi reads, so that we could label the sub-images. In total, we label 100,000 sets of sub images with half of them containing SNPs. We also vary the HiFi read coverage from the original 100x to 40x, 60x and 80x to see the impact of improvement on various coverages. To evaluate the sub-image classification results, we use the common metrics precision, recall and f-score.

3.1.5 Impact of parameters in adapter detection

As discussed in section "Method", parameters of the DBSCAN algorithm and k-mer based matching approach could affect adapter removal. There are basically four parameters: diameter_size, cluster_size, k_size and identity_value, where the first two are related to DBSCAN and the last two related to k-mer based matching. To assess the impact of parameters on adapter detection, since no ground truth of the existing adapters is available, we insert adapters into the human (HG002) HiFi reads in the way discussed in section "Training and implementation details". We detect the inserted adapters by JiuCuo with various settings of the parameters. We vary the diameter_size and cluster_size from 200 to 2,000 and from 2 to 6, respectively, with disabled k-mer based matching. We also vary the k_size and identity_value from 5 to 20 and from 0 to 1, respectively, with fixed diameter_size and cluster_size. To evaluate the adapter detection results, we also use the common metrics precision and recall.

3.2 Datasets and performance measurements

3.2.1 Datasets

The HiFi reads and reference genome of *A. thaliana* are downloaded from the CNCB database with accession numbers CRR302668 and GWHBDNP00000000.1, respectively.[20] The corresponding SNP dataset is downloaded from the 1001 Genomes database with accession number 6909. The HiFi reads and reference genome of D. melanogaster are downloaded from the NCBI database with accession numbers SRR10188371 and GCF_000001215.4, respectively. The HiFi

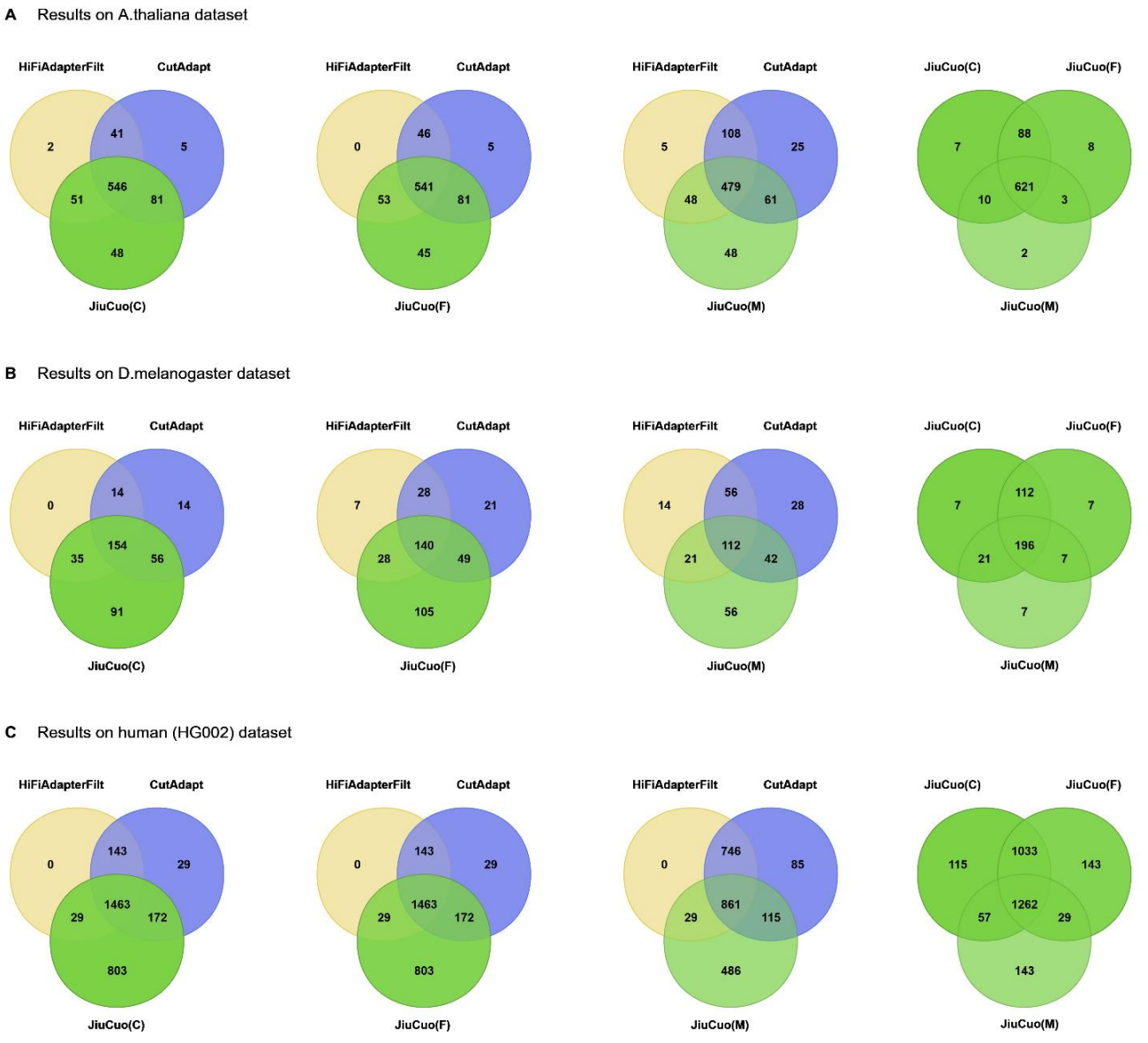

Fig. 3: Adapter removal results by various methods in Venn diagrams on (A) *A. thaliana*, (B) *D. melanogaster* and (C) human (HG002) datasets.

reads and reference genome of human (HG002) are downloaded from the NCBI database with accession numbers PRJNA586863 and GCF 000001405.40, respectively.

3.2.2 Performance measurements

We use Hifieval to evaluate the base error correction results. The evaluation metrics are as follows. True positive rate (TPR): percentage of correctly corrected bases over count of base errors existing in initial reads. False negative rate (FNR): percentage of not corrected bases over count of base errors existing in initial reads. Over-correction rate (OCR): percentage of incorrectly corrected bases over total count of corrected bases. These metrics do not take into consideration corrected reads aligned differently from the corresponding initial reads to the reference contigs, so some additional metrics are also included for completeness as follows. Position difference rate (PDR): permille of corrected reads aligned to a position of the same chromosome and homologue but different from the corresponding initial reads. Homologue difference rate (HDR): permille of corrected reads aligned to a position of the same chromosome but different homologue from the corresponding initial reads. Chromosome difference rate (CDR): permille of corrected reads aligned to a position of different chromosome from the corresponding initial reads.

We use QUAST to evaluate the assembly results. The evaluation metrics are as follows. #Contigs: number of assembled contigs. NGA50 (mbp): based on contig alignment to reference genome, the length for which the collection of all alignment blocks of that length or longer covers at least half the reference genome. Genome fraction: percentage of the reference genome covered by aligned contigs. #Misassemblies per 100 mbp: number of misassemblies in the contigs per 100 mbp of the reference genome. #Mismatches per 100 kbp: number of mismatches in the contigs per 100 kbp of the

reference genome. #Indels per 100 kbp: number of insertions and deletions in the contigs per 100 kbp of the reference genome.

In addition, we use common metrics recall, precision and f-score, and their definitions are as follows. Recall: number of detected true positive SNPs (or adapters) over the total number of SNPs (or adapters). Precision: number of detected true positive SNPs (or adapters) over the total number of detected SNPs (or adapters). F-score: $\frac{2 \cdot precision \cdot recall}{precision+recall}$.

3.3 Results

3.3.1 Comparison with existing error correction methods

Table 1(a)-(c) lists the error correction results for *A. thaliana*, *D. melanogaster* and human (HG002) datasets, respectively. On the *A. thaliana* dataset, compared with HiCanu^c, JiuCuo(C) can achieve comparable OCR ($\leq$ 5%) but much larger TPR. HiCanu^c's performance is consistent with the paper.[16] Compared with Hifiasm^c, JiuCuo(F) can achieve both comparable TPR and OCR. Though JiuCuo's base error correction results are comparable to Hifiasm^c, its corrections are essentially complementary to Hifiasm^c, and can improve the assembly quality as presented in section "Reassembly with JiuCuo" below. Compared with mdBG^c, JiuCuo(M) can achieve comparable TPR but much smaller OCR. This is because mdBG^c does not need SNP-aware error correction in its design. To further investigate the FNR and OCR changes along genome for the different methods, we divide *A. thaliana* genome into 10 kbp bins, calculate the FNRs and OCRs for each bin, and plot the FNRs and OCRs by various methods along part of the genome. Fig. S4 presents FNRs and OCRs by various methods along part of *A. thaliana* genome. Fluctuations of the FNR and OCR appear along the genome, but are within a limited range. On the *D. melanogaster* and human (HG002) datasets, the results are similar to those on the *A. thaliana* dataset. These results are obtained using Hifieval with Hifiasm assembled contigs, and the results with HiCanu contigs are listed in Table S1. Overall, these results indicate JiuCuo is effective in SNP-aware error correction achieving high TPR, low OCR and low FNR. Specifically, the low OCR demonstrates JiuCuo can distinguish base errors and SNPs, and does not falsely converting SNPs from one haplotype to another, and the high TPR demonstrates JiuCuo can largely reduce base errors.

3.3.2 Comparison with other adapter removal methods

Fig. 3(A)-(C) presents adapter removal results in Venn diagrams on *A. thaliana*, *D. melanogaster* and human (HG002) datasets, respectively. On the *A. thaliana* dataset, compared with HiFiAdapterFilt and CutAdapt, JiuCuo can detect and remove 479-546 common adapters in consistency with them. These numbers are relatively large compared with all the detected adapters by each method, indicating JiuCuo has high detection precision. In addition, JiuCuo can detect and remove 45-48 unique adapters not detected by either HiFiAdapterFilt or CutAdapt. These numbers are also relatively large compared with the unique adapters detected by each method, indicating JiuCuo has high detection recall. Using different preassembled contigs for adapter removal, JiuCuo can detect and remove 624-709 common adapters with 2-8 unique adapters. This also indicates JiuCuo has high adapter detection precision. The different adapter detection results using different preassembled contigs are due to indels and misassemblies of the contigs, which are noises in the consensus information as described in section "Method". To further investigate adapter locations along reads, we divide each read of *A. thaliana* into 20 bins, record number of adapters detected by HiFiAdapterFilt and JiuCuo(F) in each bin, and plot the distribution of adapters along reads. Since CutAdapt does not report specific adapter location in a read, it cannot be included in the plot. Fig. S5 presents distribution of detected adapters by HiFiAdapterFilt and JiuCuo(F) along *A. thaliana* reads. Blue portion of each bar represents number of adapters detected by both methods in the corresponding bin. It can be seen that HiFiAdapterFilt and JiuCuo(F) have a large number of shared adapters, while JiuCuo(F) can detect more additional ones, indicating JiuCuo has both high recall and precision. On the *D. melanogaster* and human (HG002) datasets, the results are similar to those on the *A. thaliana* dataset. The number of detected adapters on the *D. melanogaster* is smaller than the *A. thaliana* dataset, and this is probably due to differences of library preparations and specific sequencing machines. Overall, these results indicate JiuCuo is effective in adapter removal achieving high precision and recall.

3.3.3 Reassembly with JiuCuo

Table 2(a)-(c) presents assembly results with and without JiuCuo correction on *A. thaliana*, *D. melanogaster* and human (HG002) datasets, respectively. On the *A. thaliana* dataset, with JiuCuo, the number of assembled contigs decreases from 116-209 to 85-103, number of misassemblies per 100 mbp decreases from 52-140 to 49-114, number of mismatches per 100 kbp decreases from 9.24-15.68 to 8.00-13.13, number of indels per 100 kbp decreases from 2.61-5.71 to 2.39-4.93. These indicate JiuCuo can improve the genome assembly in accuracy. Besides, Fig. 4 presents MUMmer plots of the assembled contigs by Hifiasm and JiuCuo(F)+Hifiasm. As shown in the plots, the assembled contigs by JiuCuo(F) +Hifiasm have much smaller count, but can be aligned to the reference genome with higher quality. On the *D. melanogaster* dataset, JiuCuo's improvements are mostly similar to those on the *A. thaliana* dataset, while with JiuCuo, the NGA50 increases from 3.09-7.53 mbp to 7.22-12.55 mbp. These indicate JiuCuo can improve the genome assembly not only in accuracy but also completeness. Fig. 5 presents visual results of assembled contigs by Hifiasm and JiuCuo(F)+Hifiasm. The results are generated from MUMmer alignment, with each white block as a not aligned reference genome region, each black or gray block as an aligned reference genome region,

Table 2: Assembly results with and without JiuCuo correction on *A.thaliana*, *D.melanogaster* and human (HG002) datasets.

Method	#Contigs	NGA50 (mbp)	Genome fraction	#Misassemblies per 100 mbp	#Mismatches per 100 kbp	#Indels per 100 kbp
(a) *A. thaliana* dataset						
HiCanu	165	8.57	99.88%	52.35	9.24	2.61
JiuCuo(C)+HiCanu	85	8.64	99.90%	48.61	8.00	2.39
Hifiasm	209	12.75	99.96%	67.30	15.24	4.08
JiuCuo(F)+Hifiasm	96	12.75	99.93%	61.32	11.55	2.67
mdBG	116	6.07	97.78%	139.84	15.68	5.71
JiuCuo(M)+mdBG	103	6.33	98.22%	114.41	13.13	4.93
(b) *D. melanogaster* dataset						
HiCanu	1,521	7.53	93.76%	198.99	295.04	78.49
JiuCuo(C)+HiCanu	1,174	12.55	93.95%	89.06	286.03	69.83
Hifiasm	333	3.09	90.34%	123.15	397.26	88.54
JiuCuo(F)+Hifiasm	334	9.05	94.24%	103.67	285.43	62.40
mdBG	655	5.40	90.18%	183.68	294.01	207.65
JiuCuo(M)+mdBG	558	7.22	92.10%	161.42	156.42	172.03
(c) human (HG002) dataset						
HiCanu	568	76.19	96.93%	106.44	86.38	115.92
JiuCuo(C)+HiCanu	541	81.32	97.34%	102.47	82.66	109.97
Hifiasm	423	88.85	97.51%	125.47	93.85	134.63
JiuCuo(F)+Hifiasm	403	93.65	97.93%	120.41	85.53	127.78
mdBG	938	64.74	96.36%	189.39	132.93	188.03
JiuCuo(M)+mdBG	850	75.13	97.17%	170.29	126.80	186.57

Note: JiuCuo(C)+HiCanu, JiuCuo(F)+Hifiasm and JiuCuo(M)+mdBG represent HiCanu, Hifiasm and mdBG reassemblies with JiuCuo(C), JiuCuo(F) and JiuCuo(M) corrected HiFi reads, respectively.

and each switch between black and gray blocks as a contig end or misassembly. As shown in the results, the assembled contigs by JiuCuo(F)+Hifiasm are more continuous and complete. On the human (HG002) dataset, JiuCuo's improvements are also very similar to those on the *A. thaliana* dataset, while with JiuCuo, the NGA50 increases from 64.74-88.85 mbp to 75.13-93.65 mbp. Overall, these results indicate JiuCuo is effective in improving quality of genome assemblies in multiple aspects

3.3.4 Impact of model improvement in SNP-aware error detection

Table 3 lists sub-image classification results with and without improvement in Inception-v4 on *A. thaliana* dataset. Generally, with JiuCuo's improvement, values of all the metrics precision, recall and f-score can increase to some extent. Though these increments are moderate, they are important in base error correction, because the metrics are calculated with sets of sub-images and each set may contain many errors or SNPs. When the read coverage is relatively low, JiuCuo's improvement is more explicit compared with high coverage. This is obvious, since with limited information, improvement of method may affect results more than sufficient information. Overall, JiuCuo's improvement in Inception-v4 is effective in SNP-aware error detection.

3.3.5 Impact of parameters in adapter detection

Fig. 6 presents adapter detection results with various parameter settings on human (HG002) dataset. With a fixed cluster_size, increment of the diameter_size results in increased detection precision and decreased recall. This is because a large cluster diameter can put sufficient SVs into clusters but may also have some adapters clustered with SVs. With a fixed diameter_size, increment of the cluster_size results in increased detection recall and decreased precision. This is because a large cluster_size requirement can avoid having adapters clustered with SVs but may also not put sufficient SVs into clusters. Optimal results can be obtained when diameter_size=600 and cluster_size=3 inconsistency with JiuCuo's default parameter settings. In addition, with a fixed

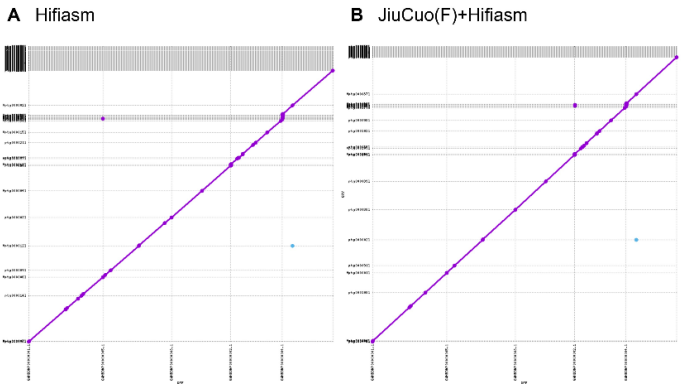

Fig. 4: MUMmer plots of assembled *A.thaliana contigs* by (A) Hifiasm and (B) JiuCuo(F)+Hifiasm.

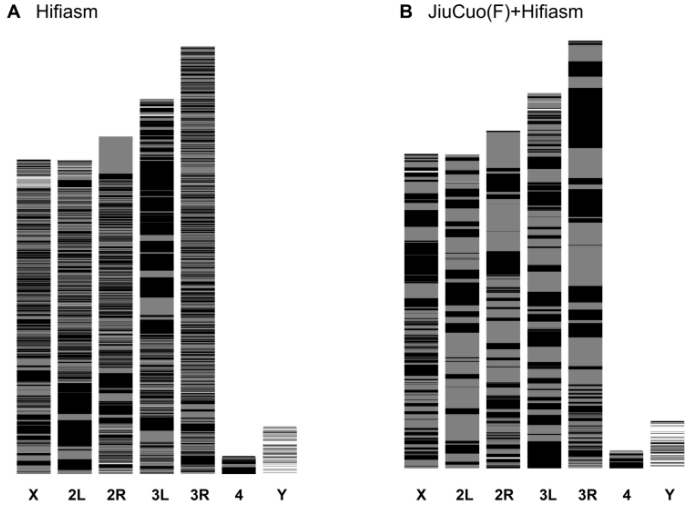

Fig. 5: Visual results of assembled *D. melanogaster* contigs by Hifiasm and JiuCuo (F)+Hifiasm. (A) *D. melanogaster* contigs by Hifiasm. (B) *D. melanogaster* contigs by JiuCuo(F)+Hifiasm.

k_size, increment of the identity_value results in increased detection precision and decreased recall. This is because a large identity_value can avoid mismatching to adapter template but may also lose some adapter matches to the template. With a fixed identity_value, increment of the k_size also results in increased detection precision and decreased recall. This is because a large k-mer can avoid mismatching to adapter template but may also lose some adapters matches to the template. Optimal results can be obtained with identity_value=0.6 and k_size=5 in consistency with JiuCuo's default parameters. Overall, JiuCuo's default parameter settings are reasonable and effective.

4. Conclusion

4.1 Advantages of this study

In this study, we propose JiuCuo, a PacBio HiFi read correction method using preassembled contigs based on deep image processing. Following HALC[7] and FLAS,[8] this is the third PacBio long read correction tool designed by us. To achieve accurate all-versus-one HiFi read alignment, JiuCuo relies on de novo contigs preassembled from initial HiFi reads as input.

Table 3: Sub-image classification results with and without improvement in Inception-v4 on *A. thaliana* dataset.

Coverage	Method	Precision	Recall	F-score
40×	Inception-v4	93.58%	90.92%	92.23%
	JiuCuo	93.97%	94.93%	94.45%
60×	Inception-v4	94.35%	94.48%	94.41%
	JiuCuo	94.71%	95.85%	95.28%
80×	Inception-v4	97.28%	96.92%	97.10%
	JiuCuo	97.87%	98.24%	98.05%
100×	Inception-v4	98.89%	98.94%	98.91%
	JiuCuo	98.92%	99.17%	99.04%

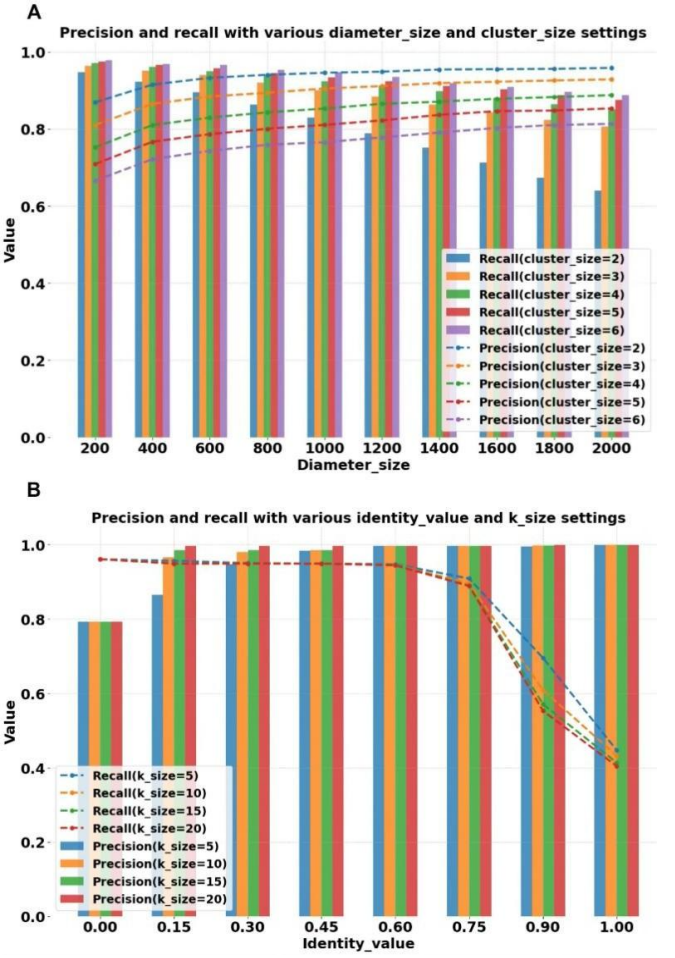

Fig. 6: Adapter detection results with various (A) diameter_size and cluster_size settings and (B) identity_value and k_size settings on human (HG002) dataset.

To distinguish errors from SNPs and to remove adapters, JiuCuo converts aligned HiFi reads into images, and makes SNP- aware error correction and adapter removal with modified deep image processing models. These are two major contributions of this study, and in experiments, the comparisons with existing error correction and adapter removal methods, as well as reassembly of corrected reads demonstrate JiuCuo is effective in correcting base errors, removing adapters and improving the HiFi read assembly. Additionally, JiuCuo contains some dedicated designs including improvement of Inception-v4 model and

combination of YOLO-v8 model with DBSCAN and k-mer based matching. These are additional contributions, and tests on impact of model improvement in SNP-aware error correction and impact of parameters in adapter detection demonstrate JiuCuo's model improvement and parameter settings are reasonable.

4.2 Disadvantages and future work

Though JiuCuo is designed in context of HiFi read assembly, it is not designed from HiFi read assembly. The problem solved in the study is solely "HiFi read correction", rather than "HiFi read assembly". There are three reasons for this choice. First, there has been a standalone validation tool, Hifieval, for HiFi read base correction regardless of downstream applications, and also standalone adapter removal tools, HiFiAdapterFilt and CutAdapt. Second, HiFi read correction itself has boarder downstream applications such as structural variant, tandem repeat, copy number variation detections. Third, the HiFi read assembly is highly difficult and complex, and can only be addressed by limited top research groups in the world. Therefore, in the experiments, we validate JiuCuo's base correction using the hifieval tool, and validate JiuCuo's adapter removal by comparing with HiFiAdapterFilt and CutAdapt. We extend the tests to reassembly of JiuCuo corrected reads in, but this is not primary part of the experiments.

In the future, we will optimize JiuCuo with more comprehensive analysis of downstream assemblies, further investigate internal relationship between JiuCuo's correction and the assembly performance, and test the assembly quality with more metrics, *e.g.* switch errors in haplotype phasing. We will also study how to effectively combine preassembled contigs from different assemblers to improve correction accuracy. In addition, we plan to incorporate spatial information into generated images, *e.g.* position of each base within a read, to further improve accuracy of JiuCuo in both base error correction and adapter removal.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 62172028).

Conflict of Interest

There is no conflict of interest.

Supporting Information

Applicable.

CRediT Statement

Jiwen Liu: software, testing, writing. **Mingfei Pan**: software, testing, writing. **Hongbin Wang**: software, testing, writing. **Hang Zhang**: methodology, writing, supervision. **Ergude Bao**: methodology, writing, supervision.

References

[1] H. Cheng, G.T. Concepcion, X. Feng, H. Zhang, H. Li, Haplotype-resolved de novo assembly using phased assembly graphs with hifiasm, *Nature Methods*, 2021, **18**, 170–175, doi: 10.1038/s41592-020-01056-5.

[2] S. Nurk, B.P. Walenz, A. Rhie, M.R. Vollger, G.A. Logsdon, R. Grothe, K.H. Miga, E.E. Eichler, A.M. Phillippy, S. Koren, HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants from high-fidelity long reads, *Genome Research*, 2020, **30**, 1291–1305, doi: 10.1101/gr.263566.120.

[3] B. Ekim, B. Berger, R. Chikhi, Minimizer-space de Bruijn graphs: Whole-genome assembly of long reads in minutes on a personal computer, *Cell Systems*, 2021, **12**, 958–968.e6, doi: 10.1016/j.cels.2021.08.009.

[4] A. Bankevich, A.V. Bzikadze, M. Kolmogorov, D. Antipov, P.A. Pevzner, Multiplex de Bruijn graphs enable genome assembly from long, high-fidelity reads, *Nature Biotechnology*, 2022, **40**, 1075–1081, doi: 10.1038/s41587-022-01256-6.

[5] M. Rautiainen, S. Nurk, B.P. Walenz, G.A. Logsdon, D. Porubsky, A. Rhie, E.E. Eichler, A.M. Phillippy, S. Koren, Telomere-to-telomere assembly of diploid chromosomes with Verkko, *Nature Biotechnology*, 2023, **41**, 1474–1482, doi: 10.1038/s41587-023-01861-x.

[6] G. Baid, D.E. Cook, K. Shafin, T. Yun, F. Llinares-López, Q. Berthet, A. Belyaeva, A. Töpfer, A.M. Wenger, W.J. Rowell, DeepConsensus improves the accuracy of sequences with a gap-aware sequence transformer, *Nature Biotechnology*, 2023, **41**, 232–238, doi: 10.1038/s41587-022-01592-7.

[7] E. Bao, L. Lan, HALC: High throughput algorithm for long read error correction, *BMC Bioinformatics*, 2017, **18**, 1–12, doi: 10.1186/s12859-017-1518-3.

[8] E. Bao, F. Xie, C. Song, D. Song, FLAS: fast and high-throughput algorithm for PacBio long-read self-correction, *Bioinformatics*, 2019, **35**, 3953–3960, doi: 10.1093/bioinformatics/btz198.

[9] R. Poplin, P.-C. Chang, D. Alexander, S. Schwartz, T. Colthurst, A. Ku, D. Newburger, J. Dijamco, N. Nguyen, P.T. Afshar, A universal SNP and small-indel variant caller using deep neural networks, *Nature Biotechnology*, 2018, **36**, 983–987, doi: 10.1038/nbt.4235.

[10] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, **31**, doi: 10.1609/aaai.v31i1.11233.

[11] R. Varghese, M. Sambath, Yolov8: A novel object detection algorithm with enhanced performance and robustness, *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, 2024, 1–6.

[12] A. Stergiou, R. Poppe, Adapool: Exponential adaptive pooling for information-retaining downsampling, *IEEE Transactions on Image Processing*, 2022, **32**, 251–266, doi: 10.1109/TIP.2022.3142418.

[13] A. Stergiou, R. Poppe, G. Kalliatakis, Refining activation downsampling with SoftPool, *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, 10357–10366, doi: 10.1109/ICCV48922.2021.01015.

[14] H. Li, Minimap2: pairwise alignment for nucleotide

sequences, *Bioinformatics*, 2018, **34**, 3094–3100, doi: 10.1093/bioinformatics/bty191.
[15] S.B. Sim, R.L. Corpuz, T.J. Simmonds, S.M. Geib, HiFiAdapterFilt, a memory efficient read processing pipeline, prevents occurrence of adapter sequence in PacBio HiFi reads and their negative impacts on genome assembly, *BMC Genomics*, 2022, **23**, 157, doi: 10.1186/s12864-022-08408-0.
[16] Y. Guo, X. Feng, H. Li, Evaluation of haplotype-aware long-read error correction with hifieval, *Bioinformatics*, 2023, **39**, btad631, doi: 10.1093/bioinformatics/btad631.
[17] M. Martin, Cutadapt removes adapter sequences from high-throughput sequencing reads, *EMBnet.journal*, 2011, **17**, 10–12, doi: 10.14806/ej.17.1.200.
[18] A. Gurevich, V. Saveliev, N. Vyahhi, G. Tesler, QUAST: quality assessment tool for genome assemblies, *Bioinformatics*, 2013, **29**, 1072–1075, doi: 10.1093/bioinformatics/btt086.
[19] G. Marçais, A.L. Delcher, A.M. Phillippy, R. Coston, S.L. Salzberg, A. Zimin, MUMmer4: A fast and versatile genome alignment system, *PLoS Computational Biology*, 2018, **14**, e1005944, doi: 10.1371/journal.pcbi.1005944.
[20] B. Wang, X. Yang, Y. Jia, Y. Xu, P. Jia, N. Dang, S. Wang, T. Xu, X. Zhao, S. Gao, High-quality Arabidopsis thaliana genome assembly with nanopore and HiFi long reads, *Genomics, Proteomics & Bioinformatics*, 2022, **20**, 4–13, doi: 10.1016/j.gpb.2021.04.009.